CLAIMS

What is claimed is:

1. A method comprising:

detecting a first location-independent error within an area of a cache memory;

determining whether the first error is a second consecutive error associated with the area; and

preventing further use of the area if the error is determined to be the second consecutive error associated with the area.

2. The method of claim 1, wherein the area comprises a cacheline of an n-way set associative static random access memory.

3. The method of claim 2, wherein the location-independent error comprises an error checking and correcting (ECC) based error.

4. The method of claim 1, further comprising:

comparing a current operation address corresponding to the first error with a stored address corresponding to a previous error; and

identifying the first error as the second consecutive error if the current operation address matches the stored address.

5. The method of claim 4, further comprising:

storing the current operation address corresponding to the first error in place of the stored address corresponding to the previous error if the current operation address does not match the stored address.

6. The method of claim 5, further comprising:

accessing the current operation address corresponding to the first error a second time to determine whether the second consecutive error occurs.

7. The method of claim 1, wherein the area comprises a cache line and wherein preventing further use of the area further comprises:

selecting the cache line; and

modifying a cache management system to at least inhibit subsequent access to the selected cache line.

8. The method of claim 7, wherein modifying a cache management system comprises modifying a value corresponding to a particular set such that the cache line is less likely to be accessed than at least one other cache line upon one or more further cache accesses by a processor.

9. The method of claim 7, wherein modifying a cache management system comprises assigning a disable state to the cache line.

10. The method of claim 9, wherein the disable state is assigned as part of a MESI state assignment.

11. The method of claim 1, further comprising:

determining whether data stored in the area of cache memory has been modified as compared to data stored in a corresponding area of main memory;

facilitating recovery of the modified data stored in the area of cache memory; and

preventing further use of the area of the cache memory after the modified data stored in the area of cache memory has been recovered.

12. A method comprising:

detecting a first error within a cache line of a cache memory;

disabling the cache line of the cache memory; and

dynamically modifying a cache management system to at least inhibit subsequent access to the disabled cache line by a processor.

13. The method of claim 12, wherein modifying a cache management system comprises modifying a value stored in a register corresponding to the cache line and associated with implementation of a least recently used algorithm, such that the cache line is less likely to be accessed than at least one other cache line within the cache memory upon one or more further cache accesses by a processor.

14. The method of claim 12, wherein modifying a cache management system comprises assigning a disable state to the cache line.

15. The method of claim 12, wherein the first error is detected via programmable built-in self test (PBIST) logic.

16. A processor comprising:

a cache memory;

error detection logic coupled to the cache memory, the error detection logic equipped to

detect a first location-independent error within an area of the cache memory,

determine whether the first error is a second consecutive error associated with the area; and

prevent further use of the area if the error is determined to be the second consecutive error associated with the area.

17. The processor of claim 16, wherein the cache memory comprises an n-way set associative static random access memory (SRAM).

18. The processor of claim 16, wherein the error detection logic comprises:

error checking and correcting logic to detect an error within the cache memory;

hard error detection logic to determine whether the detected error is a hard error; and

cacheline disable logic to disable a cacheline affected by the error if it is determined that the detected error is a hard error.

19. The processor of claim 18, wherein the hard error detection logic further comprises:

an error register to store address information indicating at least the cacheline affected by the error;

comparison logic to compare stored address information with a current operating address; and

state logic to determine whether the error is a second consecutive error based upon output from the comparison logic.

20. The processor of claim 19, wherein the cacheline disable logic further comprises:

a decoder to generate a way-disable vector prevent further use of the area; and

way-select logic to select the cacheline to be disabled based upon the way-disable vector;

wherein the way-disable vector operates to modify a cache management system including at least one of a least recently used (LRU) algorithm and a MESI protocol.

21. A processor comprising:

a cache memory and

error detection logic coupled to the cache memory, the error detection logic equipped to

detect a first error within a cache line of a cache memory;

disable the cache line of the cache memory; and

dynamically modify a cache management system to at least inhibit subsequent access to the disabled cache line by a processor.

22. The processor of claim 21, wherein the error detection logic is further equipped to modify a value stored in a register corresponding to the cache line and associated with implementation of a least recently used algorithm, such that the cache line is less likely to be accessed than at least one other cache line within the cache memory upon one or more further cache accesses by a processor.

23. The processor of claim 21, wherein the error detection logic is further equipped to assign a disable state to the cache line.

24. The processor of claim 21, further comprising programmable built-in self test (PBIST) logic coupled to the error detection logic to facilitate detection of the first location-independent error during a startup routine of the processor.

25. A system comprising:

a dynamic random access memory; and

an integrated circuit coupled to the dynamic random access memory, the integrated circuit including a cache memory and error detection logic, wherein the error detection logic is equipped to

detect a first location-independent error within an area of the cache memory,

determine whether the first error is a second consecutive error associated with the area, and

prevent further use of the area if the error is determined to be the second consecutive error associated with the area.

26. The system of claim 25, wherein the integrated circuit further includes a central processing unit and at least one input/output module coupled to the central processor unit.

27. The system of claim 25, wherein the integrated circuit is a microprocessor.

28. A system comprising:

a dynamic random access memory; and

an integrated circuit coupled to the dynamic random access memory, the integrated circuit including a cache memory and error detection logic, wherein the error detection logic is equipped to

detect a first error within a cache line of the cache memory;

disable the cache line of the cache memory; and

dynamically modify a cache management system to at least inhibit subsequent access to the disabled cache line by a processor.

29. The system of claim 28, wherein the integrated circuit further includes a central processing unit and at least one input/output module coupled to the central processor unit.

30. The system of claim 28, wherein the integrated circuit is a microprocessor.